UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/976,931	10/11/2001	Clifford L. Hersh	PA1951US	2047

22830 7590 02/15/2007

CARR & FERRELL LLP
2200 GENG ROAD
PALO ALTO, CA 94303

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2195	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	02/15/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	09/976,931	HERSH, CLIFFORD L.	
	Examiner	Art Unit	
	Lewis A. Bullock, Jr.	2195	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 January 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-7,10,13,15 and 17-27* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☒ Claim(s) *17* is/are allowed.

6)☒ Claim(s) *1-7,10,13,15,18-20 and 22-27* is/are rejected.

7)☒ Claim(s) *21* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-3, 5-7, and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Complexity of Layered Binary Search Trees with Relaxed Balance" by LARS JACOBSEN et al.

As to claim 1, JACOBSON teaches a method of reducing the number of times a tree data structure is rebalanced comprising the steps of: allowing a sub-tree of the tree data structure to grow until a number of unbalanced levels reaches a threshold and rebalancing the tree data structure when the threshold is reached (via performing relaxed balance wherein rebalancing is performed when the monitored path lengths exceed some limit) (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to

bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added.

As to claim 5, JACOBSON teaches a method of deferring the rebalancing of a tree data structure comprising the steps of: allowing a sub-tree of the tree data structure to grow to an unbalanced length; and rebalancing the tree data structure when the unbalanced length of the sub-tree reaches a threshold level (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been

equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be

greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added.

As to claim 13, JACOBSON teaches a system for deferring the rebalancing of a tree data structure comprising:. a memory for storing the tree data structure; and a processor coupled to the memory, the processor operable to track the performance of operations upon the tree data structure and rebalance the tree data structure when a number of unbalanced levels within a sub-tree of the tree data structure reaches a threshold (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of

parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. In addition, it is well known in the art and obvious to one of ordinary skill in the art that the system of JACOBSON has a processor that rebalances the tree as the limit is exceeded.

As to claims 2 and 3, JACOBSON teaches rebalancing the tree when the tree growth exceeds a threshold (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to

bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any constant value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. In addition, Official Notice is taken in that functions, such as random and logarithmic, are used to determine a number and would be obvious to one of ordinary skill in the art that this number is the threshold used for starting the rebalancing.

As to claims 6 and 7, refer to claims 2 and 3 for rejection.

3. Claims 4, 10, 15, 18-20 and 22-27 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Complexity of Layered Binary Search Trees with Relaxed Balance" by LARS JACOBSEN et al. in view of "Relaxed AVL Trees, Main-Memory Databases, and Concurrency" by OTTO NURMI et al.

As to claim 10, JACOBSON teaches a method of performing a rebalancing operation upon a tree data structure comprising the steps of: allowing a sub-tree of the tree data structure to grow unbalanced to a threshold level; developing, in the case where the sub-tree reaches the threshold level, rebalancing operation tasks (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed

structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first set and a second set.

NURMI teaches the rebalancing operations comprise a first and second set of rebalancing operation tasks operable to effect a first and second set of element state transitions respectively; performing the first set of operation tasks in a first phase (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value); and performing the second set of operation tasks in a second phase (via rotating the tree to put the tree in balance) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset

such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claim 15, JACOBSON teaches a system comprising: means for storing a tree data structure; means for tracking the execution of operations upon the tree data structure; and means for rebalancing the tree data structure when an unbalanced sub-tree of the tree data structure reaches a threshold level (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing

is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first phase and a second phase.

NURMI teaches the rebalancing operations including a first rebalancing phase in which rebalancing operations are executed in parallel and nodes of the unbalanced sub-tree are unlocked (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value), and a second rebalancing phase in which different rebalancing operations are executed (via rotating the tree to put the tree in balance) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed

balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claim 18, JACOBSON teaches a method of deferring the rebalancing of a tree data structure comprising the steps of: tracking the performance of operations upon the tree data structure; and rebalancing the tree data structure when an unbalanced sub-tree of the tree data structure reaches a threshold level (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should

belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first phase and a second phase.

NURMI teaches the rebalancing further comprising executing simultaneous rebalancing operations on the tree data structure including performing any first phase

operation task of each of the simultaneous rebalancing operations in a first phase using parallel processes (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value), developing a set of serial rebalancing operations (rotation operations) during the first phase, and performing any second phase operation tasks of each of the simultaneous rebalancing operations in a second phase, the second phase operation task having at least one of the set of serial rebalancing operations (via rotating the tree to put the tree in balance) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claims 19 and 20, JACOBSON teaches a method of rebalancing a tree data structure, the method comprising: allowing a sub-tree of the tree data structure to grow unbalanced until a threshold level is reached (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor

path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first set and a second set.

NURMI teaches the rebalancing operations include a first set of rebalancing operation task, the first set of operation tasks operable in parallel on one or more unlocked nodes of the tree data structure during a first phase of the rebalancing (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value); developing a second set of rebalancing operation tasks during execution of the first set of rebalancing operation tasks; and executing the second set of rebalancing operation tasks during a second phase of the rebalancing (via rotating the tree to put the tree in balance) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation

or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). It is inherent to the teachings of NURMI that the rotation operations executing in the second phase do not navigate to the nodes, because they were already traversed in the first phase in determining whether an unbalance exists. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claims 22-24, JACOBSON teaches a method of rebalancing a tree data structure, the method comprising: allowing a sub-tree of the tree data structure to grow unbalanced until a threshold level is reached (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the

tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first set and a second set.

NURMI teaches the rebalancing operations comprise executing a first set of rebalancing operation tasks during a first rebalancing phase, the first rebalancing phase being characterized by navigation between nodes of the sub-tree (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value); and executing a second set of rebalancing operation tasks during a second rebalancing phase, the second rebalancing phase including navigation to two or more nodes of the sub-tree, the navigation being independent of pointers between nodes of the sub-tree (via rotating the tree to put the tree in balance) (pg. 10, "Each

invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). It is inherent to the teachings of NURMI that the first rebalancing operations are performed in parallel on unlocked nodes of the tree since these tasks are described without locking (pg. 3 1st paragraph). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claims 25 and 26, JACOBSON teaches a method of maintaining a tree data structure, the method comprising: allowing the tree data structure to grow unbalanced and rebalancing the tree when a limit or other condition has occurred (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update

without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first set and a second set.

NURMI teaches the rebalancing operations comprise performing a first set of rebalancing operation tasks during a first rebalancing phase on a plurality of nodes in the tree data structure, the first set of rebalancing operation tasks being configured for execution while the plurality of nodes are unlocked and for insertion and deletion of nodes (via decreasing the tag value for inserted or to be deleted nodes during the rebalancing process that was determined to be unbalanced based on the tag value); and performing a second set of rebalancing operation tasks on the plurality of nodes in a second rebalancing phase, the second set of rebalancing operation tasks being different than the first set of rebalancing operation tasks and being configured for further operations on the plurality of nodes, the second rebalancing phase occurring after completion of the first rebalancing phase (via performing rotations to balance the tree) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). It is inherent to the teachings of NURMI that the first rebalancing operations are performed in parallel on

unlocked nodes of the tree since these tasks are described without locking (pg. 3 1st paragraph). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claim 27, JACOBSON teaches a method comprising: storing a tree data structure; tracking the execution of operations upon the tree data structure; and rebalancing the tree data structure when an unbalanced sub-tree of the tree data structure reaches a threshold level (pg. 2, 2nd paragraph, "informally relaxed balance is a term used for the following. If a search tree has been equipped with relaxed balance, the searching and updating have been uncoupled from the rebalancing. Thus it is now possible to search and make an update without performing any rebalancing. For this to be well-defined, the balance constraints must be weakened (relaxed!) in such a way that the tree after an update is still in the now broader class of trees. Additionally, the standard tree, which is made relaxed, should belong to the class, and the overall goal of the (presumably generalized and/or expanded) collection of rebalancing operations is to bring the tree back to fulfilling the constraints of the standard balanced tree on which the relaxed version is based...In a sequential system, bursts of requests, possibly from an external source, can be served faster if rebalancing is "turned off" during the period. After the burst, rebalancing should gradually bring the tree back in balance, while request are served at the same time. In a parallel (shared memory) system, a naïve implementation would lock the root of the tree so frequently that the degree of

parallelism would be extremely low. In relaxed structures, it is generally possible to exclusively lock only nodes which will be involved in pointer changes, instead of all nodes which might be involved in pointer changes...The cost of the relaxation is that the guaranteed worst-case bound of logarithmic path lengths is temporarily lost. The options are to trust that this does not become a problem for these short periods of time (maybe the requests are known to be close to uniform), to monitor path lengths and rebalance when some limit is exceeded..."). It would be obvious to one of ordinary skill in the art that since the rebalancing is relaxed that the user can set the threshold to any value and thus be greater than one, otherwise if the threshold is one there is no relaxing of the rebalancing, it would occur as nodes are added. However, JACOBSON does not explicitly detail that the rebalancing operations comprise a first set and a second set.

NURMI teaches the rebalancing including a first rebalancing phase in which rebalancing operations are executed in parallel (via adjust the tag values of the manipulated nodes) and nodes of the unbalanced sub-tree are unlocked, and a second rebalancing phase in which different rebalancing operations are executed (via rotating the nodes) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's

relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the imbalance does not propagate up the tree.") (See also pg. 10-15). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

As to claim 4, NURMI teaches the rebalancing operations comprise a first and second set of rebalancing operation tasks operable to effect a first and second set of element state transitions respectively; performing the first set of operation tasks in a first phase (via decreasing the tag-value during the rebalancing process that has found an unbalanced node based on the tag-value); and performing the second set of operation tasks in a second phase (via rotating the tree to put the tree in balance) (pg. 10, "Each invocation of the process performs some local operations in the tree that not only decrease the unbalance of the tree, but also preserve the relaxed balance of the tree. This effect is achieved by adjusting the tag values of the manipulated nodes appropriately and by using rotations...we divide the rebalancing step into two phases. In the first phase, the absolute value of the tag of the child is decreased by one, and the tag value of p is reset such that p's relaxed height is preserved. Decreasing the absolute value of the tag of a node changes the node's relaxed height; it may introduce a balance factor of 2 or –2 at the node's parent. Such a balance factor is changed in the second phase by a rotation or a tag adjustment; the tags are reset such that the

imbalance does not propagate up the tree.") (See also pg. 10-15). Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of JACOBSON with the teachings of NURMI in order to facilitate updating a relaxed tree without locking (pg. 3, 1st paragraph).

Allowable Subject Matter

4. Claim 17 is allowed.

5. Claim 21 is objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Response to Amendment

6. The 131 affidavit filed on January 23, 2007 under 37 CFR 1.131 has been considered but is ineffective to overcome the "Complexity of Layered Binary Search Trees with Relaxed Balance" reference.

7. The LARS JACOBSEN reference is a statutory bar under 35 U.S.C. 102(b) and thus cannot be overcome by an affidavit or declaration under 37 CFR 1.131. The examiner previously provided, and has submitted another copy herein, of the cited reference which has a disclosure date of November 1999, from the University of Southern Denmark, a little less than 2 years from the day the application was filed. The cited document was pre-printed on the University's electronic database in 1999 wherein a weekly mailing list was sent out that the document was available to be accessed (see

attached copy of document with information on the database and mailing list wherein the document was added in 1999). Therefore, the reference is a statutory bar reference and cannot be overcome by the 131 affidavit. Since there are no other arguments in the application, the rejections are maintained.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (571) 272-3759. The examiner can normally be reached on Monday-Friday, 8:30 a.m. - 5:00 p.m..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

February 9, 2007

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER